(2)

February 1990

UILU-ENG-90-2206
CSG-120

COORDINATED SCIENCE LABORATORY
College of Engineering

FAILURE ANALYSIS AND MODELING OF A MULTICOMPUTER SYSTEM

Sujatha Srinivasan Subramani

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) (CSG-120) UILU-ENG-90-2206	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois; 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research NASA
6c. ADDRESS (City, State, and ZIP Code) 1101 W. Springfield Ave. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) NASA Ames Research Center N244-7 Moffett Field, CA 94035; Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION NASA JSEP; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER NASA NCA 2-184, NASA Ames NCA 2-301, and N00014-84-C-0149 (JSEP)
8c. ADDRESS (City, State, and ZIP Code) NASA Ames N244-7 Moffett Field; Arlington, VA 22217	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

"Failure Analysis and Modeling of a Multicomputer System" (Master's thesis)

12. PERSONAL AUTHOR(S)
Subramani, Sujatha Srinivasan

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ______ TO ______	1990 February	38

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	failure analysis, VAX cluster, error data, reliability, availability models, reward analysis

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This thesis describes the results of an extensive measurement-based analysis of real error data collected from a 7-machine DEC VaxCluster multicomputer system. In addition to evaluating basic system error and failure characteristics, we develop reward models to analyze the impact of failures and errors on the system. The results show that, although 98% of errors in the shared resources recover, they result in 48% of all system failures. The analysis of rewards shows that the expected reward rate for the Vax Cluster decreases to 0.5 in 100 days for a 3-out-of-7 model, which is well over a 100 times that for a 7-out-of-7 model. A comparison of the reward rates for a range of k-out-of-n models indicates that the maximum increase in reward rate (0.25) occurs in going from the 6-out-of-7 model to the 5-out-of-7 model. The analysis also shows that software errors have the lowest reward (0.2 vs. 0.91 for network errors). The large loss in reward rate for software errors is due to the fact that a large proportion (94%) of software errors lead to failure. In comparison, the high reward rate for network errors is due to fast recovery from a majority of these errors (median recovery duration is 0 seconds).

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

FAILURE ANALYSIS AND MODELING OF A MULTICOMPUTER SYSTEM

BY

SUJATHA SRINIVASAN SUBRAMANI

B.Engr., Anna University, 1984

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1990

Urbana, Illinois

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

THE GRADUATE COLLEGE

DECEMBER 14, 1989

WE HEREBY RECOMMEND THAT THE THESIS BY

SUJATHA SRINIVASAN SUBRAMANI

ENTITLED FAILURE ANALYSIS AND MODELING OF A MULTICOMPUTER SYSTEM

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF MASTER OF SCIENCE

R. K. Iyer
Director of Thesis Research

[illegible]
Head of Department

Committee on Final Examination†

____________________ Chairperson

† Required for doctor's degree but not for master's.

O-517

ABSTRACT

This thesis describes the results of an extensive measurement-based analysis of real error data collected from a 7-machine DEC VaxCluster multicomputer system. In addition to evaluating basic system error and failure characteristics, we develop reward models to analyze the impact of failures and errors on the system. The results show that, although, 98% of errors in the shared resources recover, they result in 48% of all system failures. The analysis of rewards shows that the expected reward rate for the VaxCluster decreases to 0.5 in 100 days for a 3-out-of-7 model, which is well over a 100 times that for a 7-out-of-7 model. A comparison of the reward rates for a range of k-out-of-n models indicates that the maximum increase in reward rate (0.25) occurs in going from the 6-out-of-7 model to the 5-out-of-7 model. The analysis also shows that software errors have the lowest reward (0.2 vs. 0.91 for network errors). The large loss in reward rate for software errors is due to the fact that a large proportion (94%) of software errors lead to failure. In comparison, the high reward rate for network errors is due to fast recovery from a majority of these errors (median recovery duration is 0 seconds).

ACKNOWLEDGMENTS

I will forever be grateful to my adviser Prof. Ravi Iyer for his excellent guidance and encouragement. I thank him very much.

Thanks are due to Matt Hood of NASA Ames, Robin Sahner of Motorola Inc., and Stan Kerr of the Computing Services Office, each of whom was very helpful at many stages of this thesis. I would also like to thank all my friends at the Center for Reliable and High Performance Computing for their friendship, help and support. In particular, I would like to thank Dong Tang, Bob Dimpsey, Luke Young, Inhwan Lee and Rene Llames who made many valuable suggestions to this work.

Very special thanks to my wonderful family who make everything possible for me: my parents, my husband, my sister and her family, and my in-laws.

TABLE OF CONTENTS

1. INTRODUCTION

This thesis describes the results of an extensive measurement-based analysis of real error data collected from a 7-machine DEC VaxCluster system at the NASA Ames Research Center. The measurements encompasses a period of about 8 months of continuous operation from December 1987 to August 1988. The analysis evaluates basic system characteristics such as frequency breakdown of errors and failures, mean times between errors and failures and effectiveness of error recovery. Analysis of reward models is performed to quantify the impact of failures for different k-out-of-n models of the VaxCluster. A comparison of these models is performed to estimate the impact of different VaxCluster configurations on the reward rate. Reward models are also used to analyze the impact of different error types on each system.

The results show that shared resources (network and disks) are a major reliability bottleneck in the system. Although the system recovers from 98% of errors in the shared resources, these result in 48% of all failures. This is due to the high frequency of errors (93%) in the shared resources. The average Mean Time Between Failures (163 hours) is about 100 times the average Mean Time Between Errors (1.6 hours). The study of recovery durations indicates that CPU errors have the fastest recovery time (mean of 12 seconds) and I/O errors have the slowest recovery time (mean of 400 seconds). Analysis of reboot data shows that about 30% of reboots occur in conjunction with multiple error events, although multiple error events span only 0.8% of the measured duration. Only 24% of the these reboots lead to recovery.

The analysis of rewards shows that the expected system reward rate decreases to 0.5 in 100 days for a 3-out-of-7 model, which is well over a 100 times that for a 7-out-of-7 model. A comparison of the reward rates for a range of k-out-of-n models indicates that the maximum increase in reward rate (0.25) occurs in going from the 6-out-of-7 model to the 5-out-of-7 model. The analysis also shows that software errors have the lowest reward (0.2 vs. 0.91 for network errors). The low reward rate for software errors is due to the fact that a large proportion (94%) of software errors lead to failure. In comparison, the high reward rate for network errors is due to fast recovery from a majority of these errors (median recovery duration is 0 seconds).

The next section discusses related research. Chapter 2 introduces the measured system and measurements made. Chapter 3 describes the error classification and the results of the prelimnary data analysis. Chapter 4 develops reward models to analyze the impact of failures and errors on the system. Chapter 5 highlights the major results of this study and makes suggestions for future work.

1.1 Related Research

Analytical models for hardware failures have been extensively investigated in the literature [6,7,16,17,23]. In many of these studies, the time between failures is usually assumed to be exponentially distributed although time-dependent failure rates and graceful degradation have been considered. Availability and performability issues have been extensively studied in [7,16,23]. An overview of numerical approaches for computation of instantaneous and cumulative measures for Markov reward models is discussed in [18].

Measurement-based analyses of computer system failures have also evolved significantly in the last decade. The preliminary research in this area was an analysis of system-wide failures at the Stanford Linear Accelerator Computer Facility [2,10,11]. The analysis showed that the average system failure rate correlated strongly with the average workload on the system. Similar results based on measurements on the DEC systems were reported in [3]. The effect of workload imposed stress on software were investigated in [12]. Later, systematic methods which used error and usage measurements to model the impact of system activity on reliability were developed [13]. In [15], it was shown that undetected software-related errors were commonly due to specification errors, implementation errors or logic errors. In [14], data from both IBM and Cyber mainframes were used to develop a methodology for recognizing symptoms of persisting errors.

The analytical and the measurement-based techniques are brought together in [8] where a joint resource-usage/error/recovery model using error and resource usage data collected from an IBM system is constructed. The model provides detailed information on system behavior under normal and error conditions. A significant result of this reliability/performability model is that a semi-Markov process, as opposed to a Markov process, is better to model the system behaviour.

Most of the above studies have been on single machines. A study of a distributed system, at this stage, would be valuable in furthering our understanding of failure behavior in large and complex configurations. The purpose of this thesis is to discuss the results of an extensive measurement-based analysis of real error data collected from a 7-machine DEC VaxCluster multicomputer system. The next section describes the Vax architecture with specific reference to error detection, measurement and logging.

2. MEASUREMENT ENVIRONMENT

The measured system is a seven machine VaxCluster, a distributed system consisting of 7 VAX-11's and 4 mass storage controllers connected by the CI (Computer Interconnect) bus. An intelligent hardware interface called the CI port connects each node (computer or controller) and the CI bus. The CI bus is physically organized as a star topology, at the center of which is a coupler connecting all nodes through radial CI paths. The key features of the VaxCluster are: separate processors and memories connected by a message-oriented interconnect running instances of the same copy of the distributed VAX/VMS operating system, shared physical access to disk storage, and high-speed memory-to-memory block transfers between nodes [13].

Exceptions in the VaxCluster

There are a number of methods to handle exceptions in a VaxCluster. Exceptions are triggered by software consistency checks or by the hardware and detect both software and hardware errors. There are three types of exceptions, namely, aborts, faults and traps. An abort is the most severe form of an exception condition. When an instruction is aborted, the machine registers and memory may be left in an indeterminate state and hence it is not possible for recovery to occur. Faults leave the machine registers and memory in a consistent state. Once the fault is eliminated, the instruction may be restarted but correct results cannot be guaranteed. A trap is the least severe form of exception. The machine registers and memory are consistent and the address of the next instruction to execute is stored on the machine stack. The process can be restarted with the same state as before the trap occurred.

Errors in the VaxCluster

The VAX/VMS operating system maintains a log of a variety of normal and abnormal events including detailed error data. Errors are captured by the hardware detection circuits or by software consistency checks. Errors may originate in various locations including the disk subsystems, channels, tape drives, network devices peripheral processors, central processor and main memory. Each time the system detects an error, the relevant information is recorded in a system file called ERRORLOG. This study utilizes the ERRORLOG data between December 9, 1987 and August 16, 1988 on the seven machines in the VaxCluster. The machines Earth, Europa, Jupiter, Leo, Mars, Mercury, and Saturn were operating continuously during the measurement period except for brief periods when the machines were brought down for repair or preventive maintenance.

A sample of the fields logged by the system and used in this study is shown in Figure 2.1. Each entry of the error log file contains considerable information on the nature of the error and includes the error entry number, the date and time of the entry, the system identification, the device or subsystem on which the error occurred, the contents of the device registers and different levels of device dependent data pertaining to the error. The following sample of the error data shows some of the different error types logged by different machines[1].

The data contained information on both "errors" and "failures". For the purpose of this study we define an error as a detection of an abnormailty in any one of the machines. If an error led to a loss of service for that machine, it is defined as a failure. The system auto-reboot log was used to identify failures. An auto reboot is an action taken by the system to reload the console software in an attempt to recover from existing system problems. It causes an interruption of service since no processing can be done during the time a reboot is in progress.

For the purposes of this study, the errors were classified into the following five error classes according to the subsystems or devices on which they occurred.

[1] Here the entry numbers are not consecutive; the sample is intended to provide a feel for the different types of errors that were logged.

Entry	System ID	Logging Time	Subsystem	Interpretation
62	Earth	11-NOV-1987 15:41:32.64	I/O, H1$DUA51:	Disk drive error
139	Earth	11-NOV-1987 17:39:17.15	CI, EAR$PAA0:	Path #0 went from good to bad
141	Earth	11-NOV-1987 17:39:17.17	CI, EAR$PAA0:	Software is closing virtual circuit
200	Earth	12-NOV-1987 18:49:27.50	I/O, H3$MUA1:	Tape drive error
2007	Earth	30-NOV-1987 14:58:13.89	BugCheck	Unexpected system service exception
3260	Mercury	24-DEC-1987 04:54:52.06	Memory, TR #2	Corrected memory error
3264	Mercury	24-DEC-1987 04:55:28.89	Memory, TR #2	Corrected memory error
1846	Europa	17-JAN-1988 12:44:11.29	BugCheck	Bad memory deallocation
11796	Mars	7-FEB-1988 10:22:31.40	CPU, MBA	Adapter power-down
10939	Jupiter	1-APR-1988 09:57:39.40	Unknown Device	
14209	Jupiter	16-MAY-1988 13:37:04.97	CPU, SBI	Unexpected read data fault

Sample error log fields
Figure 2.1

1. CPU — CPU related errors. Examples: "cache parity", "translation buffer parity", "SBI (Synchronous Backplane Interconnect) fault";

2. Software — software errors. Examples: "unexpected system service exception", "bad memory deallocation request size or address", "kernel stack not valid";

3. I/O — disk, drive, and controller errors;

4. Network — bus and port errors. Examples: "Software is losing virtual circuit", "Data cable state changes from good to bad";

5. Memory — memory ECC errors;

Any of the error classes can lead to automatic reboots of the system, which is counted as downtime because no processing can be done when a reboot is in progress. The next section discusses the processing of the error logs that was necessary to read the data.

Data Processing

The statistical analysis package SAS [21] was used to read the error log files and process the data. SAS is a software system for data analysis that provides many statistical routines ranging from basic descriptive statistics to complex time series analysis.

The error log files were available in VMS backup binary format. The following steps had to be carried out for each of the seven errorlog files to convert the data to a format that could be processed on the IBM mainframe. First, the ANALYZE utility provided by VMS was used to convert each of the seven errorlog files to ASCII format. The ASCII version of each file was approximately 200 Megabytes. Then, the headers from the ASCII files were stripped in order to have a form acceptable to SAS. Each of the seven files were converted to an EBCDIC format and written to tape that could be read on an IBM mainframe. The next chapter presents results of preliminary analyses and provides an understanding of the various error characteristics.

3. SYSTEM AND ERROR CHARACTERIZATION

This chapter provides analysis and statistics that give a preliminary understanding of the error behavior of the VaxCluster. Error coalescing is done to merge multiple records relating to the same problem. Basic system characteristics such as frequency breakdown of errors and failures, mean times between errors and failures and error recovery are evaluated.

The results show that shared resources (network and disks) are a major reliability bottleneck in the system. Although 98% of errors in the shared resources recover, they result in 48% of all system failures. This is due to the high frequency of errors (93%) in the shared resources. The average Mean Time Between Failures (163 hours) is about 100 times the average Mean Time Between errors (1.6 hours). The study of recovery durations indicates that CPU errors have the fastest recovery time (mean of 12 seconds) and I/O errors have the slowest recovery time (mean of 400 seconds). Analysis of reboot data shows that about 30% of reboots occur in conjunction with multiple error events, although multiple error events span only 0.8% of the measured duration. Only 24% of the these reboots lead to recovery.

Error Breakdown

Recall that the error records were divided into five error classes i.e., CPU, memory, I/O, network and software errors. The error frequency for each class is given in Table 3.1. The table shows that there is a wide variation in the errors across the machines. For example, over 20% of all logged errors occurred on Mercury, while Earth contributed only 8% of the errors. Looking at the different error types, we see that I/O errors are dominant (nearly 80%); software errors have the lowest frequency (less than 0.5%). We also see a wide variation in the number of I/O and CPU errors recorded on each of the machines. Saturn has over 14000 I/O errors while Earth has about 5000. For CPU errors, the maximum frequency is on Earth (84) while the minimum is on Leo (0).

Error Events

As with other error studies of this type [10,23], a single problem commonly led to many repeated error observations occurring in rapid succession. In order to ensure that the analysis is not biased by repeated observations of the same problem, an algorithm that coalesces all error entries which had the same error type and occurred within a 5 minute interval of each other was used. The coalesced errors represented by a single record was called an error event. Thus, an error event represents the occurrence of one or more errors of the same type in rapid succession and is defined by the number of errors in the event and by the time span of the event itself. The algorithm for forming the error events is shown below.

```
IF <time from previous error is less than or equal to 5 minutes>
  AND <type of device affected> = <type of device affected in previous error>
  AND <device unit number> = <device unit number in previous error>
    THEN <fold error into event being built>
    ELSE <start a new event>.
```

Frequency of Error Classes										
Class	Earth	Europa	Jupiter	Leo	Mars	Mercury	Saturn	Total	Average	%
CPU	84	27	59	0	59	3	38	270	38	0.34
I/O	5525	9380	9113	7536	8152	8161	14122	61989	8855	79
Network	1133	1296	987	1356	1008	1255	1152	8187	1169	10.4
Memory	1	2	10	1311	0	6598	3	7925	1132	10
Software	35	26	28	12	40	37	40	218	31	0.27
Total	6929	10822	10230	9272	10244	16066	15369	78932		
%	8.7	13.7	12.96	12.97	11.74	20.35	19.47			

Breakdown of raw errors
Table 3.1

Overall Frequency					
Machine	Raw	%	Events	%	%change
MERCURY	16066	20.35	4131	14.02	74.28
SATURN	15369	19.47	4130	14.01	73.12
EUROPA	10822	13.7	4141	14.05	61.73
LEO	10244	12.97	4439	15.06	56.88
JUPITER	10230	12.96	4207	14.28	58.87
MARS	9272	11.74	4603	15.62	50.35
EARTH	6929	8.7	3809	12.92	45.02
All	78932	100	29460	100	62

Overall Frequency
Table 3.2

Table 3.2 highlights the extent to which the coalescing of errors reduces the data and compensates for multiple recordings that bias the raw data. The table shows that there is a 62% reduction in errors; the number of errors is reduced from approximately 79000 to 30000. The raw error count is reduced by varying degrees on each machine, ranging from a high of 74% on Mercury to a low of 45% on Earth. The relative contribution of each machine is between 13% to 15% after coalescing as opposed to 9% to 20% for the raw (uncoalesed) data. It is clear that the coalesced errors thus provide a balanced insight into the error characteristics of the system.

Error Clusters					
Class	Raw	%	Events	%	%reduction
CPU	270	.34	166	.56	38.5
I/O	61989	79	23556	80.78	61.9
Network	8187	10.4	3676	12.60	55
Memory	7925	10	1555	5.33	80.3
Software	218	.27	206	.7	5.5

Error Clusters
Table 3.3

Table 3.3 shows the reduction due to coalescing for each of the error classes. A comparison with the raw error counts is also given. It can be seen that the major contribution is from I/O errors that constitute 80% of the error events. The errors in the shared resources, I/O errors and network errors, account for 93% of all events. Next we analyze the failures caused by each of the error clases.

Failure due to errors

The number of failures that occurred is identified by the number of automatic system reboots. The number of failures that were caused by each error type is shown in Table 3.4. Of the total of nearly 30000 error events, about 390 lead to failure i.e., (i.e., approximately 1% of events). Thus 99% of the errors recover without reboot or system shut down. Although software events constitute only 0.7% of error events, about 94% of all software events led to system failures. In comparison, 3.5% *of the 166 CPU events lead to system failure.*

Table 3.4 also shows that about 2% of all the errors in the shared resources (I/O and network errors) led to failures. These, however, resulted in more than 48% of all failures. This indicates that although the shared resources are generally robust, i.e., the system is able to

Number of Failures		
Class	% fail	Number
Software	94.4	195
CPU	3.5	6
I/O	.5	117
Network	1.6	73
Memory	0	0

Number of failures
Table 3.4

recover from the vast majority of errors in them, they still constitute a major reliability bottleneck simply due to their sheer numbers. So an improvement in the VAXcluster reliability will necessitate an even higher error coverage in the shared resources. This may require the design of an ultra reliable network to reduce the raw error rate, not just the recoverability.

Time between errors and failures

The time between errors and failures statistics are given in Table 3.5. The average MTBF is over 100 times the average MTBE (164/1.56). There is not a significant variation in the MTBE statistics across the machines. The MTBF, however, varies considerably across the machines. The largest one (278 hours for Leo) is about 5 times the smallest one (52 hours for Earth). The maximum error free time periods vary from 1.32 days to 24 days for the seven machines, whereas the maximum time interval in which there was no failure varies from 35 to 61 *days across the seven machines.*

Table 3.6 shows the relationship between network and I/O errors. In looking down the two columns at the maximum time between network and I/O errors for each of the seven machines, we see that a strong correlation exists. The longer a machine functions without a network error, the longer it functions without an I/O error.

Error Type	Earth	Europa	Jupiter	Leo	Mars	Mercury	Saturn	Average
Mean TBE (hour)	1.47	1.71	1.68	1.55	1.31	1.59	1.62	1.56
Mean TBF (hour)	52.1	137.2	153.6	277.9	203.2	166.7	156.8	163.9
Max TBE (day)	11.75	11.84	1.32	1.56	4.39	24.07	2.52	8.21
Max TBF (day)	36.63	34.91	48.76	47.04	34.95	43.97	61.23	43.93

Time between errors and failures
Table 3.5

I/O and network errors (TBE in hours)		
Machine	Net(Max TBE)	I/O (Max TBE)
JUP	31.60	119.52
LEO	41.6	137.2
Sat	60.5	156.4
Mars	105.4	180.2
Earth	285.4	306
Europa	285.4	308.4
Mercury	577	597

I/O and network errors (TBE in hours)
Table 3.6

Error recovery duration and densities

The span of an error event provides information on the time taken by the system to recover from a particular error. The number of points that are found in each event also provides a measure of the density of error events. The rank statistics of the spans (error recovery duration) and density of events formed for each error type is given for each machine in Appendix B. The tables indicate that the maximum values of the spans are far greater than their corresponding means. This indicates that recovery is instantaneous in most cases, but there are a few cases with large recovery times. For example, the maximum span of I/O errors (6532 seconds) is approximately sixteen times the value of the mean (397 seconds); the span of 75% of CPU errors is zero seconds (instantaneous recovery).

Table 3.7 shows the mean durations of the events for the different error types. The data shows that the fastest recovery is for CPU errors (12 seconds) and the slowest recovery is for I/O errors (400 seconds). Table 3.7 also shows the mean densities (number of points in a event) for the different error types. It is seen that I/O events contain the maximum number of points

Cluster Statistics				
	Cluster Means		Largest Clusters	
Class	Duration (in secs.)	Points	Duration (in secs.)	Points
CPU	12.38	3.85	288	28
I/O	397.18	50.77	6532	1111
Network	50.26	2.48	1179	30
Memory	0	1.5	0	12
Software	0	1.05	0	2

Cluster Statistics
Table 3.7

(50) and that network errors contain the minimum number of points (2). Clearly the overheads due to redundant detection and logging of the same error is highest in the case of I/O errors. Restriction of multiple recordings of I/O is likely to improve performance. Now that we have studied recovery from error events, we proceed next to study groups of error events that lead to multiple error events in the system.

Multiple error events

The study of error events provided insight into the ocurrance and recovery of the various errors in the VaxCluster. In this section we study error events occurring very close to each other and thereby resulting in high internal error rates in the system. Error events that occur simultaneously or in rapid succession (less than five minutes apart) were grouped into multiple error events. Such multiple error events can invoke different recovery mechanisms in succession and hence may impose considerable overhead on the system.

About 4000 (14.5%) error events fall within multiple error events. The total holding time of multiple error events is 44 hours (0.8% of the total measurement period). Table 3.8 shows the percentage of error events of each type that fall into multiple error events. It is seen that

30% of system reboots occurred in periods with multiple error events (i.e., among error events that lasted for a total of 0.8% of the total measurement period). A significant proportion of network errors (36%) and CPU errors (27%) are multiple error events. Of all the I/O error events (which contribute over 80% of all error events), only 11% are multiple errors.

Table 3.9 highlights the recovery characteristics of multiple error events and reboots. We see from Table 3.9 that multiple memory errors have highest recovery rate (50%) and lead to no failures. Multiple software errors have the lowest recovery rate (9%) and the highest failure rate (91%). Given the severity of software errors (94% of all software errors lead to failure), this is not surprising.

It is seen fron Table 3.9 that 37% of network errors and 27% of I/O errors recover in the presence of multiple errors. In comparison, the overall recovery rate for network and I/O error events is high (98% for network and 99% for I/O). The 3.7% of multiple I/O errors that lead to failure account for 85% of all failures caused by I/O errors. The 4.9% of multiple network

Multiple Error Events	
Error type	Percentage
Network	36%
Reboots	30%
CPU	27%
Software	20%
Memory	11.5%
I/O	11.4%

Multiple Error Events
Table 3.8

Multiple Error Event Recovery		
Error type	Recovery	Failure
Memory	50%	0
Network	37%	4.9%
I/O	27%	3.7%
Reboot	24%	
CPU	13%	13%
Software	9%	91%

Multiple Error Event Recovery
Table 3.9

errors that lead to failures account for 88% of all system failures caused by network errors.

In studying the recovery characteristics of CPU errors, we see that 13% of *multiple* CPU errors lead to recovery and 13% lead to failure. All system failures caused by CPU errors were caused by multiple CPU errors. Only 24% of the reboots that were caused by multiple error events lead to recovery, providing furthur evidence that recovery from multiple errors is complex and frequently unsuccessful.

This chapter studied error events for each of the error classes and failures caused by some of the error events. Recovery durations were quantified and compared. Multiple error events were studied. We saw specific modes of propagation of errors. The next chapter deals with the impact of errors on the performance of the VaxCluster.

4. REWARD MODELING AND ANALYSIS

This chapter consists of two sections. In the first section reward models are developed to analyze the impact of failures for different k-out-of-n configurations of the VaxCluster. A comparison of these models is performed to estimate the impact of different VaxCluster configurations on the reward rate. In the second section a reward model is developed to analyze the impact of different error types on each system.

The analysis of rewards shows that the expected system reward rate decreases to 0.5 in 100 days for a 3-out-of-7 model, which is well over a 100 times that for a 7-out-of-7 model. A comparison of the reward rates for a range of k-out-of-n models indicates that the maximum increase in reward rate (0.25) occurs in going from the 6-out-of-7 model to the 5-out-of-7 model. The analysis also shows that software errors have the lowest reward (0.2 vs. 0.91 for network errors). The large loss in reward for software errors is due to the fact that a large proportion (94%) of software errors lead to failure. In comparison, the high reward rate for network errors is due to the fast recovery from a majority of these errors (median recovery duration is 0 seconds).

Performability of the VaxCluster based on failures of nodes.

Failure of one or more constituent nodes can cause a significant degradation in a distributed system. When a node in the VaxCluster fails, it exits from the VaxCluster and thereby the number of machines available for processing decreases. In order to investigate the effect of the failure of different nodes, a range of k-out-of-n models for the VaxCluster were analyzed. A k-out-of-n system functions if and only if at least k of the n components of it are functioning [20]. In this analysis we considered k-out-of-n models for for $k \in [3,7]$ and $n=7$, since the failure data had no more than five simultaneous failures.

Failure modeling

Initially, a Markov model to depict the different failures states of the VaxCluster was constructed. The failure model neglects all non-failure error events. We define F_i to be the state in which i machines have failed. Thus, F_0 represents the state where no machine has failed ,and, F_1 represents the state that one machine machine has failed, and so on. At any time the VaxCluster is in one of the 6 states (F_0, F_1, F_2, ... F_5). Since there were no instances of six or seven joint failures states F_6 and F_7 do not exist.

The transition probabilities for the defined 6-state model were then calculated from the measured data. Given that the system is in state S_i, the probability $P_{i,j}$ that it will go to state F_j is calculated as follows:

$$P_{i,j} = \frac{\textit{observed number of transitions from state } F_i \textit{ to state } F_j}{\textit{observed number of transitions out of state } F_i}$$

Table 4.1 shows the measured transition probability matrix for the model. We see from Table 4.1, that, given multiple failures, the chances of additional failures is significant For example, the transition probability from F_3 to F_4 is 0.53, and transition probability from F_4 to F_5 is 0.31.

State	F0	F1	F2	F3	F4	F5
F0	0.000	0.95	0.04	0.01	0.000	0.000
F1	0.83	0.000	0.14	0.03	0.000	0.000
F2	0.35	0.5	0.000	0.15	0.000	0.000
F3	0.000	0.12	0.35	0.000	0.53	0.000
F4	0.000	0.000	0.06	0.63	0.000	0.31
F5	0.000	0.000	0.000	0.000	1.000	0.000

Transition Probability for the Failure Model
Table 4.1.

State	F0	F1	F2	F3	F4	F5
Mean HT (in mins.)	1523.89	44.86	6.96	4.65	18.92	133.5
Total HT (hour)	5613.1	183.2	7.2	1.32	4.1	8.9
Occ. Prob.	0.964	.031	.001	.0002	.0007	.0015

Holding Time & Occupancy Probability for the Failure Model
Table 4.2

The mean holding time, the total holding time, and the occupancy probability for each state of the failure model are shown in Table 4.2. The highest failure state, F_5, has the longest mean holding time. This indicates that when 5 machines failed, the longest time was needed by the system to recover or repair. The occupancy probability provides strong evidence that multiple failures in high states are related. For example, the measured occupancy probability of F_5 is 0.0015, but the probability under the independent case is approximately 21×0.0065^5 which is about 6×10^6 times higher (where 0.0065 is the average failure probability of the seven machines). Even for F_2 where the difference between the two probabilities (in the real case and the independent case) is the smallest among all multiple failure states, the measured occupancy probability is still seven times that for the independent case.

Reward for the Failure Model

To investigate the impact of failures, a reward rate is defined for the failure model. For every machine that is functioning, a reward of 1/7 is awarded. When j machines function, a reward of j/7 is awarded. Thus, in the normal (non-failure) state F_0, the reward rate is 7/7 (equalling 1) because all seven machines are functioning. Similarly, when six machines are functioning, a reward of 6/7 is awarded; and so on until a reward of 3/7 is awarded when only three machines are functioning. In our model, the following five cases are considered:

Case 1 (impact of one or more failures): In the first case, referred to as a 7-Out-of-7 model, we assume that every machine is critical to system operation. Thus, the whole VAXcluster's reward rate goes to zero as soon as a single failure occurs. This gives a worst case estimate of the system reward. The reward rate for state i, F_i, is then defined as

$$r_i = \begin{cases} 1 & \text{if } i=0 \\ 0 & \text{if } i\neq 0 \end{cases}.$$

In order to examine the impact of any machines's failure on the VaxCluster, we make F_1, F_2, F_3, F_4, F_5 absorbing states.

Case 2 (impact of two or more failures): In the second case, referred to as a 6-Out-of-7 model, we calculate the reward characteristic based on the assumption that at least six machines must be fully operating for the system to do any useful work. In order to examine the impact of the failure of two machines on the VaxCluster, we make F_2, F_3, F_4, F_5 absorbing. When all 7 machine are functional, the reward rate is 1. When any six machines are functional, the reward rate is 6/7. When less than six machines are functional, the reward rate is 0.

Case 3 (impact of 3 or more failures): In case 3, referred to as a 5-out-of-7 model, we examine the impact of the failure of three machines on the VaxCluster by making F_3, F_4, F_5 absorbing. Reward rates of 1, 6/7 or 5/7, are assigned for 7, 6 or 5 functional machines Otherwise the reward rate is set to 0.

Case 4 (impact of 4 or more failures): In case 4, referred to as a 4-out-of-7 model, we examine the impact of the failure of four machines on the VaxCluster by making F_4, F_5 absorbing. Reward rates of 1, 6/7, 5/7 or 4/7 are assigned if 7, 6, 5 or 4 functional machines respectively. Otherwise the reward rate is set to 0.

Case 5 (impact of five failures): In case five, referred to as a 3-Out-of-7 model, we calculate the reward characteristics based on the assumption that at least three machines must be fully operating for the system to do any useful work. F_5 is made the absorbing state. If the number of failing machines is less than five, the system is still considered working with a degraded reward rate. Recall that the worst failure situation was five machine's simultaneous failures, so this definition gives a best estimate of the system reward. Under this consideration, the reward rate of state i, F_i, is defined as

$$r_i = j/7 = 1 - i/7 .$$

where i = 7 -j,

j being the number of functioning machines; and i being the number of failed machines. (This is the general case common to cases 1-5).

As shown in [18] and [8], given a time t within the time domain considered, the system reward rate, R(t), can be expressed as

$$R(t) = \{r_i \mid \text{system is in state i at time t}\}.$$

Therefore the expected system reward rate at time t can be evaluated as

$$E[R(t)] = \sum_i r_i p_i(t) ,$$

were p_i is the probability of the system being in state i at time t. The system accumulated reward by time t, A(t), can be derived from

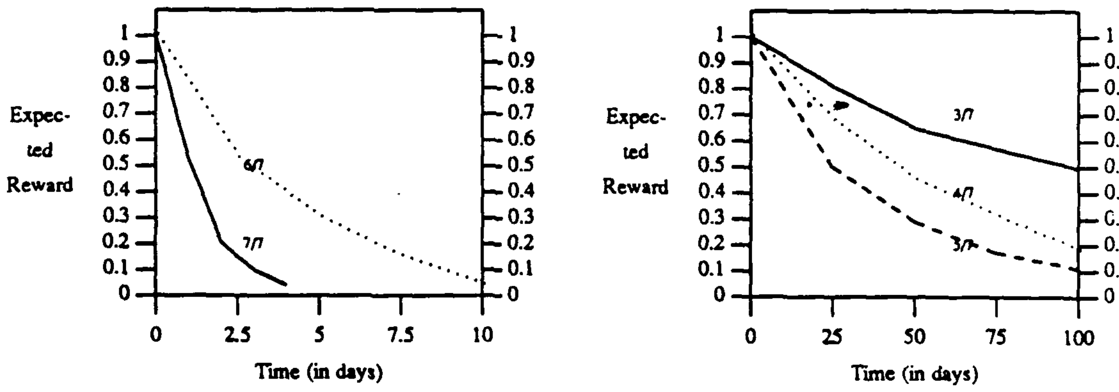
$$A(t) = \int_0^t R(x)dx .$$

Therefore the expected system accumulated reward by time t can be computed by

$$E[A(t)] = \int_0^t \sum_i r_i p_i(x)dx.$$

The performability analysis was carried out using SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator). SHARPE [22] is a modeling tool developed at Duke University that provides several model types ranging from reliability block diagrams to complex Markov models. It allows the the user to construct and analyze performance, reliability and availability models. SHARPE was used to calculate the expected reward rate.

The plots of the expected reward rate vs. time for each of the above five cases are shown in Figure 4.1[2]. A comparison of the different k-out-of-7 models is done to estimate the impact of different VaxCluster configurations on the rewards characteristics[3]. We look at the time taken to reach a fixed reward rate (0.5 in this case) by each of the models. For example, the time taken to reach an expected reward rate of 0.5 is 21 hours for the 7-out-of-7 model, Looking at the entire range of models, it is seen that maximum increase in time taken to reach a reward rate of 0.5 occurs in going from the 6-out-of-7 model to the 5-out-of-7 model (3 days vs. 25 days).



Expected Reward Rates for k-out-of-n models
Figure 4.1

[2] Two separate plots are shown because the time scales vary and we cannot make a comparison by plotting all the curves on the same graph.

[3] Note that the underlying assumption is that the 3-out-of-7 model gives the best estimate of system reward, and that the 2-out-of-7 and 1-out-of-7 models are not considered.)

Next a comparison between the different k-out-of-n models is made by observing the change in reward at a fixed reference time. We choose 50 days as the reference point. (Note that the reward rate for the 7-out-of-7 model and 6-out-of-7 model has stabilized in 50 days and so we use their steady state values in the comparison.) For example, after 50 days of operation, the expected reward rate is .65 in the 3-out-of-7 model. Looking at the entire range of models, it is seen that the maximum increase in reward rate (.25) occurs in going from the 6-out-of-7 model to the 7-out-of-7 model (.04 to .29).

As seen from the holding time in state F_0 in Table 4.2, the VaxCluster functions in its full configuration (7-out-of-7) about 96 % of the time. Therefore, the loss of nodes in the VaxCluster happens less than 5% of the time. However, the change in reward characteristics observed above serves to emphasise the fact that fewer nodes in the VaxCluster result in dramatic loss of work since repair times are not insignificant. The next section uses reward models to analyze the impact of different error types on each system.

Performance Loss in the VaxCluster due to errors

The markov model to describe the error behavior of each system is defined somewhat differently than for the failure mode. Seven states are defined: normal operation, CPU errors, memory errors, software errors, I/O errors, network errors, and system failure. Thus, each system is modeled as a seven-state Markov process. To obtain a measure of the useful work done by the system operating under error conditions, a reward is defined for each state and the expected reward rate is calculated.

Reward for the error model

In assigning rewards to each of the error states, we note that when the system enters an error state it may stay there for a measurably finite duration depending on the time span of the error event. During an error event, the system goes into repeated error-recovery cycles until the error vanishes and the loss of work is proportional to the error density (number of errors per unit time) in the current state. The higher the error density, the higher the recovery overhead and the lower the performance. Thus error density is one of the parameters that influences the system performance and is incorporated in the reward rate r_i as discussed below.

Following the reward structure used in [8], we define a reward rate r_i per unit time for each state i in the model as follows:

$$r_i = \begin{cases} \dfrac{s_i}{s_i + e_i} & \text{if } i \in S_N \cup S_F \\ 0 & \text{if } i \in S_R \end{cases} ,$$

where s_i and r_i are the service rate and the error rate in state i respectively. Thus one unit of reward is given for each unit of time that the process stays in the normal state S_N. The penalty paid depends on the number of errors generated by an error event. With an increasing number of errors the penalty per unit time increases, and accordingly, the reward rate decreases. Zero reward is assigned to recovery states. This is due to the fact that during the recovery process the system does not contribute any useful work toward the system performance besides recovering from an error.

The expected reward rate $E[X(t)]$ can be evaluated as:

$$E[X(t)] = \sum_i p_i(t) r_i .$$

where $p_i(t)$ is the probability of occupying state i at time t.

SHARPE was used to carry out the reward analysis and calculate the expected reward rates. The impact of a specific error type on the expected reward is evaluated by assigning the specific state to be an absorption state. Therefore, the following cases are considered:

Case 1) with failure as the absorbing state.

Case 2) with CPU and failure as the absorbing states.

Case 3) with software and failure as the absorbing state.

Case 4) with memory and failure as the absorbing state.

Case 5) with network and failure as the absorbing state.

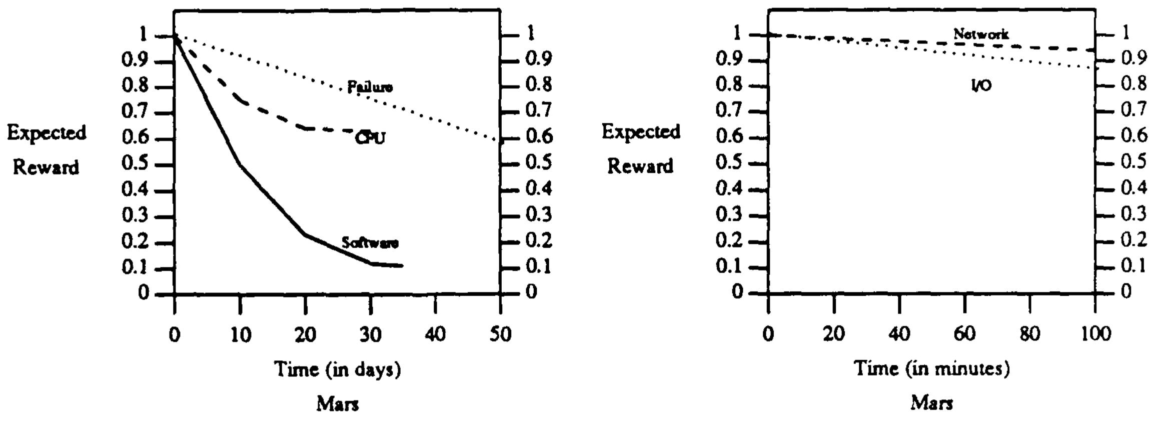
Case 6) with I/O and failure as the absorbing state.

In case 1 all errors are recovered from. In case 2 we discontinue recovering from CPU errors and therefore expect to measure the impact on the reward for a CPU error. In case 3 we recover from CPU errors but stop recovery from software errors. Thus we measure the impact on the reward for a software error. Similarly in case 4, 5, and 6 we measure the impact due to each of memory, network and I/O errors respectively by stopping recovery in turn from each of these states.

As an example, the plots for expected reward rate due to the error states in Mars are shown in Figure 4.2. It is clear that the fastest decrease in reward rate is for software errors. To illustrate this, we look at the time taken by the reward rate of software and CPU errors to decrease to 0.7. It is seen that the expected reward rate decreases to 0.7 in 7 days of operation for software errors[4] and in 16 days of operation for CPU errors. The reward characteristics for network and I/O errors are considerably different. Taking 0.95 as a reference point, the expected reward rate decreases to this value in 37 minutes for I/O errors and in 96 minutes for network errors. The plots of the expected reward rates for each of the error states on all the machines are shown in Appendix C.

Table 4.3 indicates the expected reward values for each error type on each of the machines and also the Average of the Steady State Values (ASSV) of each error type for each machine. The CPU, memory and software values are shown for 30 days of operation and the network and I/O values are shown for 100 minutes of operation (because the reward rates for I/O and

[4] The reward rate of I/O and network errors does not decrease to 0.7.



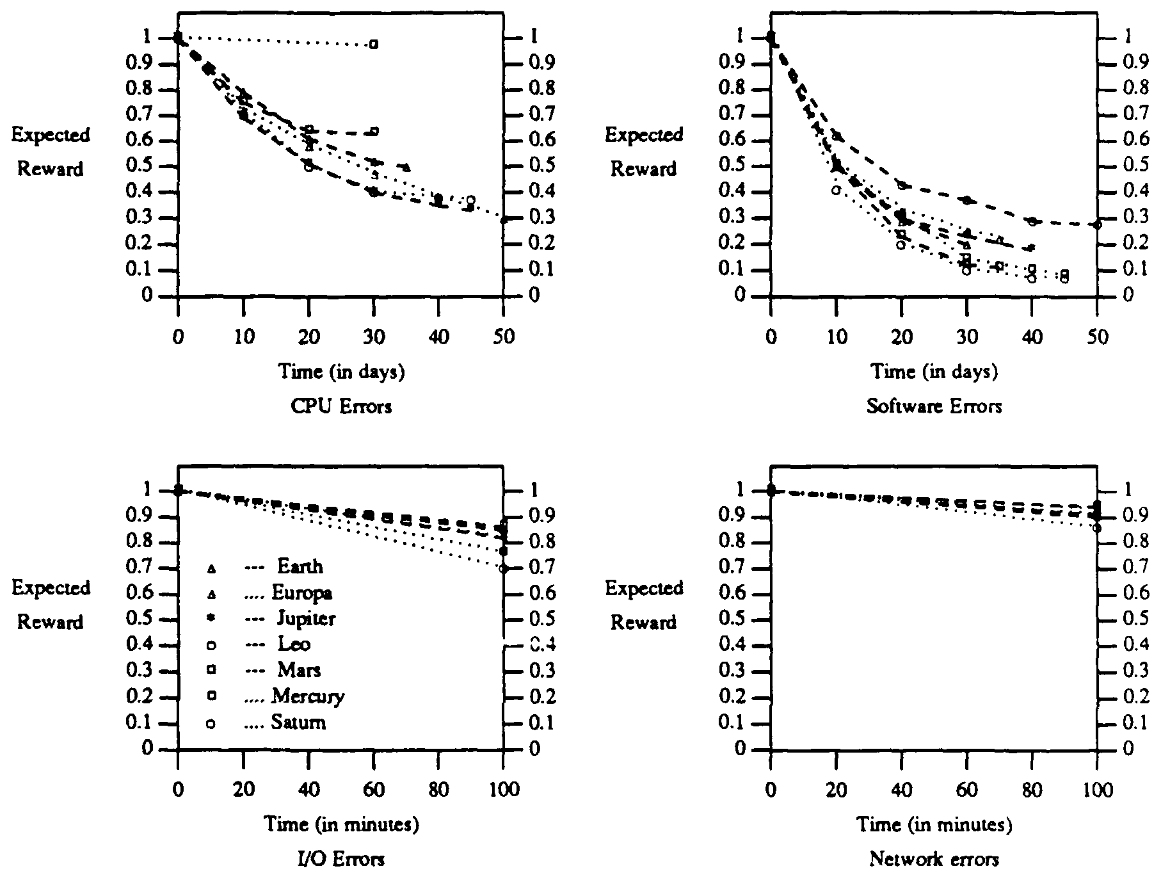
Expected Reward Rate for Mars.
Figure 4.2

	Expected Reward					
	CPU	Memory	Software	I/O	Network	ASSV
EARTH	.52	.99	.2	.86	.91	.69
EUROPA	.47	.99	.25	.82	.93	.69
JUPITER	.4	.97	.23	.82	.94	.67
LEO	1	.4	.37	.848	.9	.7
MARS	.63	1	.12	.863	.94	.71
MERCURY	.97	.21	.14	.76	.91	.59
SATURN	.44	.99	.1	.70	.86	.61
Average	.54	.792	.20	.81	.912	

Expected Reward Rate
Table 4.3

network errors reach steady state values in 100 minutes).

Comparing the average expected reward rates (shown in Table 4.3), we see that the reward rate due to software errors is the lowest (0.2) and the average expected reward rate due to net-



Expected Reward Rate By Error Type
Figure 4.3

work errors is the highest (.91). The large loss in reward for software errors is due to the fact that a large proportion of software errors (94%) lead to failure. In comparison, the high reward rate for network errors is due to fast recovery from a majority of these errors (median recovery duration is 0 seconds).

Comparing I/O and network errors (from Table 4.3), we find that the average expected reward rate of I/O errors (.81) is ten points lower than the average expected reward rate for netork errors (.91), although a lesser proportion of I/O errors (0.5%) lead to failure (vs. 1.6% for network errors). The reason for the lower reward for I/O errors is due to their large frequencies (80% of all errors are I/O errors vs. 12% for network errors) and longer recovery times (median recovery duration is 75 seconds for I/O errors and 0 seconds for network errors).

The average steady state values shown in Table 4.3 indicates close similarity for 5 machines (between .69 and .71). Therefore the average loss of work due to error states is comparable for these five machines. The average steady state values for Mercury and Saturn are about ten points lower than for the other five systems because of the large frequencies of their recoverable memory and I/O errors respectively. We see here that large frequencies of errors cause measurable loss in reward.

Figure 4.3 shows the variation in expected reward rate across machines for each error type. There is a significant variation in reward for each error type across the machines. The maximum variation is for rewards to CPU errors where the steady state values range from .3 to 1. The minimum variation is for rewards to network errors where the steady state values range from .88 to .92. This is because network errors affect all machines. Rewards for software errors vary from a value as low as .05 to .29.

5. CONCLUSIONS

In this thesis, the results of an extensive measurement-based analysis of real error data collected from a 7 machine VaxCluster multi-computer system was discussed. System error and failure characteristics were evaluated. Analysis of reward models was performed to quantify the impact of failures for different k-out-of-n models of the VaxCluster and to estimate the impact of different VaxCluster configurations on the reward rate. Reward models were also used to analyze the impact of different error types on each system.

The error entries were divided into 5 error classes comprising CPU, software, memory, I/O and network errors. It was seen that I/O and network errors constituted 94% of the total errors. Approximately 1% of errors led to failure; the system recovered from 99% of the errors without reboots or system shut down. About 94% of all software errors led to system failures. While only about 2% of all the errors in the shared resources (I/O and network errors) led to failures, they resulted in more than 48% of all failures. This indicates that although the system was able to recover from the vast majority of errors in the shared resources, they still constituted a major reliability bottleneck simply due to their large frequencies. Multiple error events that occurred in periods of high internal error rates in the system led to 30% of system reboots although the multiple error events spanned only 0.8% of the measured duration.

The analysis of rewards showed that the expected system reward rate decreased to 0.5 on 100 days of operation for the 3-out-of-7 model, which was well over a 100 times that for the 7-out-of-7 model. A comparison of the reward rates for a range of k-out-of-n models indicated that the maximum increase in reward rate (0.25) occured in going from the 6-out-of-7 model to the 5-out-of-7 model. A comparison of the reward rates for the various error clsses showed that software errors had the lowest reward (0.2) due to the fact that a large proportion (94%) of them led to system failures. Network errors had the highest reward (0.91) due to their fast recovery times (median recovery duration was 0 seconds).

Suggestions for future research

It would be very valuable to investigate other multi-computer systems in a similar fashion so that a wide range of results on the reliability and performance of computer systems is available. A second extension of this work is in the area of on-line diagnosis. Given the vast amount of data, it would be useful to implement methods to use past error data for on-line analysis and failure diagnosis.

REFERENCES

[1] Balkovich E. E., et al., "VAXcluster Availability Modeling", *Digital Technical Journal*, No. 5, pp. 69-79, Sept. 1987. 1983.

[2] Butner, S. E. and Iyer, R. K., "A Statistical Study of Reliability and System Load at SLAC", *IEEE FCTS10*, Kyoto, Japan, pp. 207-212, Oct. 1-3, 1980.

[3] Castillo, X. and Siewiorek, D. P., "A Performance-Reliability Model for Computing Systems", *IEEE FTCS-10*, Kyoto, Japan, pp. 187-192, Oct. 1-3, 1980.

[4] Castillo, X. and Siewiorek, D. P., "A Workload Dependant Software Reliability Prediction Model," *Proceedings of the 12th International Symposium on Fault Tolerant Computing*, Santa Monica, CA, pp279-285, June 22-24, 1982.

[5] Digital Equipment Corporation, *VAX/VMS System Messages and Recovery Procedures Reference Manual*, April 1986.

[6] Geist, R. M. and Trivedi, K., "Ultrahigh Reliability Prediction for Fault-Tolerant Computer Systems", *IEEE Transactions on Computers*, pp. 1118-1127, Dec. 1983.

[7] Goyal, A. and Tantawi, A. N., "Numerical Evaluation of Guaranteed Availability", *IEEE FTCS-15*, Ann Arbor, Michigan, pp. 324-329, June 19-21, 1985.

[8] Hsueh, M. C., *Measurement-Based Reliability/Performability Models*. Ph.D. Dissertation, Department of Computer Science, University of Illinois at Urbana-Champaign, August 1987.

[9] R. K. Iyer, P. Velardi, "Hardware-related software errors: measurement and analysis," IEEE Trans. Software Eng., vol. SE-11, February 1985, pp. 223-231.

[10] Iyer, R. K. and Rossetti, D. J., "A Statistical Load Dependency Model for CPU Errors at SLAC", *FTCS-12*, Santa Monica, California, pp. 363-372, June 1982.

[11] Iyer, R. K., Butner, S. E., and McCluskey, E. J., "A Statistical Failure/Load Relationship: Results of A Multicomputer Study", *IEEE Transactions on Computer, Vol. C-31, No. 7, pp. 697-706, July 1982.*

[12] Iyer, R. K., and Rossetti, D. J., "Effect of System Workload on Operating System Reliability: A Study on IBM 3081", *IEEE Transactions on Software Engineering*, Vol. SE-11, No. 12, pp. 1438-1448, December 1985.

[13] Iyer, R. K., Rossetti, D. J., and Hsueh, M. C., "Measurement and Modeling of Computer Reliability as Affected by System Activity", ACM Transactions on Computer Systems, Vol. 4, No. 3, pp. 214-237, August 1986.

[14] Iyer, R. K., Young, L. T., Sridhar, V., "Recognition of Error Symptoms in Large Systems," *Proceedings of the 1986 IEEE-ACM Fall Joint Computer Conference*, Dallas, Texas, November 2-6, 1986, pp. 797-806.

[15] Kelly, J. and Avizienis, A.A., "A Specification-Oriented Multi-Version Software Experiment", *Proceedings of the 13th International Symposium on Fault-Tolerant Computing*, Milano, Italy, pp. 120-126, June 28-30, 1983.

[16] Meyer, J. F., "Closed-Form Solutions of Performability", *IEEE Transactions on Computers*, pp. 648-657, July 1982.

[17] Ng, Y. W. and Avizienis, A. A., "A Unified Reliability Model for Fault-Tolerant Computers", *IEEE Transactions on Computers, pp. 1002-1011, Nov. 1980.*

[18] Reibman, A., Smith R., and Trivedi K., "Markov and Markov Reward Model Transient Analysis: An Overview of Numerical Approaches", *European Journal of Operational Research, Vol. 40, pp. 257-267, 1989.*

[19] Ross, S. M., *Stochastic Processes*, John Wiley & Sons, Inc., 1983.

[20] Ross, S. M., *Introduction to Probability Models*, Third Edition, Academic Press, Inc., 1985.

[21] SAS Institute Inc., *SAS User's Guide: Basics*, Version 5 Edition, Cary, NC., USA, 1985.

[22] Sahner, R.A. and Trivedi, K.S., "SHARPE: Symbolic Hierarchical Automated Reliability and Performance Evaluator," *User's Guide*, Durham, NC, September 1986.

[23] Trivedi, K., et. al., "Modeling Imperfect Coverage in Fault-Tolerant Systems", *IEEE FTCS-14*, Florida, pp. 77-82, June 20-22, 1984.

[24] Tsao, M. M., *Trend Analysis and Fault Predication*, Technical Report No. CMU-CS-83-130, Dept. of Elec. Eng. and Comp. Sci., Carnegie-Mellon University, Pittsburgh, 1983.

[25] P. Velardi, R. K. Iyer, "A study of software failures and recovery in MVS," *IEEE Transactions on Computers*, vol. C-33, July 1984, pp. 343-349.

APPENDIX A

MERCURY - Errors		
Class	Raw	%
Reboot	12	.0007
CPU	3	.0001
I/O	8161	50.79
Network	1255	7.81
Memory	6598	41.06
Software	37	.002

JUPITER - Errors		
Class	Raw	%
Reboot	33	.003
CPU	59	.005
I/O	9113	89.08
Network	987	9.6
Memory	10	.0009
Software	28	.002

SATURN - Errors		
Class	Raw	%
Reboot	14	.0009
CPU	38	.002
I/O	14122	91.88
Network	1152	7.4
Memory	3	.0001
Software	40	.002

MARS - Errors		
Class	Raw	%
Reboot	13	.001
CPU	59	.006
I/O	8152	87.92
Network	1008	10.87
Memory	0	0
Software	40	.004

EUROPA - Errors		
Class	Raw	%
Reboot	91	.008
CPU	27	.002
I/O	9380	86.6
Network	1296	11.9
Memory	2	.0001
Software	26	.002

EARTH - Errors		
Class	Raw	%
Reboot	151	2.1
CPU	84	1.2
I/O	5525	79.7
Network	1133	16.3
Memory	1	.0001
Software	35	.005

LEO - Errors		
Class	Raw	%
Reboot	29	.002
CPU	0	0
I/O	7536	73.5
Network	1356	13.2
Memory	1311	12.7
Software	12	.001

Breakdown of Errors

APPENDIX B

Time Span of Error Clusters (in Seconds) - MERCURY			
	Reboot	CPU	Memory
Mean	0	0	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	0	0	0
	I/O	Network	Software
Mean	422.88	28.85	0
Min.	0	0	0
25%	0	0	0
Median	8	0	0
75%	812	0	0
Maximum	2943	638	0

Number of points in a Cluster - SATURN			
	Reboot	CPU	Memory
Mean	1.07	11	1
Min.	1	1	1
25%	1	1.75	1
Median	1	9.5	1
75%	1	19.25	1
Maximum	2	28	1
	I/O	Network	Software
Mean	184.24	2.49	1.1
Min.	1	1	1
25%	1	1	1
Median	17	2	1
75%	301	3	1
Maximum	1111	22	2

Number of points in a Cluster - MERCURY			
	Reboot	CPU	Memory
Mean	1.16	2	3.50
Min.	1	1	1
25%	1	1	2
Median	1	2	4
75%	1	3	5
Maximum	2	3	12
	I/O	Network	Software
Mean	81.47	2.52	1.18
Min.	1	1	1
25%	1	1	1
Median	4	2	1
75%	92	3	1
Maximum	784	14	2

Time Span of Error Clusters (in Seconds) - EUROPA			
	Reboot	CPU	Memory
Mean	24.37	32.55	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	305	233	0
	I/O	Network	Software
Mean	398.96	70.36	0
Min.	0	0	0
25%	0	0	0
Median	48	0	0
75%	516	0	0
Maximum	5419	1325	0

Time Span of Error Clusters (in Seconds) - SATURN			
	Reboot	CPU	Memory
Mean	0	0	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	0	0	0
	I/O	Network	Software
Mean	980.30	48.69	0
Min.	0	0	0
25%	0	0	0
Median	439.5	0	0
75%	1460	0	0
Maximum	6352	1179	0

Number of points in a Cluster EUROPA			
	Reboot	CPU	Memory
Mean	1.16	5.18	1
Min.	1	1	1
25%	1	1	1
Median	1	4	1
75%	1	8	1
Maximum	4	14	1
	I/O	Network	Software
Mean	34.81	2.52	1
Min.	1	1	1
25%	1	1	1
Median	3	1	1
75%	25	2	1
Maximum	435	30	1

Time Span of Error Clusters (in Seconds) - LEO			
	Reboot	CPU	Memory
Mean	0	0	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	0	0	0
	I/O	Network	Software
Mean	148.98	56.78	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	40.75	1	0
Maximum	3759	1179	0

Number of points in a Cluster - JUPITER			
	Reboot	CPU	Memory
Mean	1	2.52	1
Min.	1	1	1
25%	1	1	1
Median	1	2	1
75%	1	3	1
Maximum	1	9	1
	I/O	Network	Software
Mean	29.38	2.24	1
Min.	1	1	1
25%	1	1	1
Median	3	2	1
75%	23	2	1
Maximum	359	22	1

Number of points in a Cluster - LEO			
	Reboot	CPU	Memory
Mean	1.2	0	2
Min.	1	0	1
25%	1	0	1
Median	1	0	2
75%	1	0	3
Maximum	2	0	3
	I/O	Network	Software
Mean	5.79	3.24	1.08
Min.	1	1	1
25%	1	1	1
Median	2	2	1
75%	3	4	1
Maximum	164	22	2

Time Span of Error Clusters (in Seconds) - MARS			
	Reboot	CPU	Memory
Mean	0	.338	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	0	16	0
	I/O	Network	Software
Mean	231.74	54.35	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	229	0	0
Maximum	3760	1179	0

Time Span of Error Clusters (in Seconds) - JUPITER			
	Reboot	CPU	Memory
Mean	0	23.7	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	18	0
Maximum	0	288	0
	I/O	Network	Software
Mean	319.19	52.79	0
Min.	0	0	0
25%	0	0	0
Median	30	0	0
75%	445	0	0
Maximum	5413	1178	0

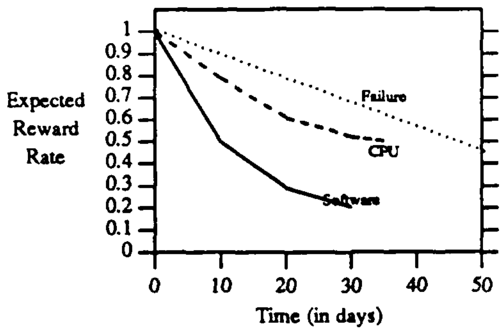
Number of points in a Cluster - MARS			
	Reboot	CPU	Memory
Mean	1	1.37	0
Min.	1	1	0
25%	1	1	0
Median	1	1	0
75%	1	1	0
Maximum	1	5	0
	I/O	Network	Software
Mean	9.59	2.24	1
Min.	1	1	1
25%	1	1	1
Median	2	2	1
75%	7	2	1
Maximum	164	22	1

Time Span of Error Clusters (in Seconds) - EARTH			
	Reboot	CPU	Memory
Mean	24.53	5.35	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	0	0	0
Maximum	357	188	0
	I/O	Network	Software
Mean	278.24	40.05	0
Min.	0	0	0
25%	0	0	0
Median	0	0	0
75%	153	0	0
Maximum	5413	965	0

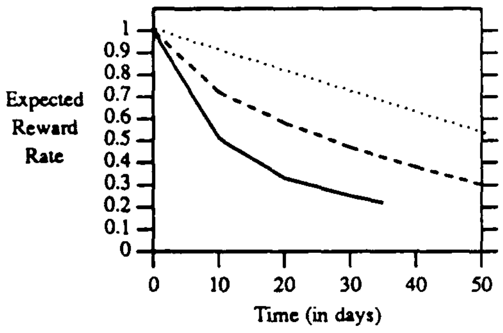
Number of points in a Cluster - EARTH			
	Reboot	CPU	Memory
Mean	1.18	1.08	1
Min.	1	1	1
25%	1	1	1
Median	1	1	1
75%	1	1	1
Maximum	4	3	1
	I/O	Network	Software
Mean	10.13	2.16	1
Min.	1	1	1
25%	1	1	1
Median	1	1	1
75%	4	2	1
Maximum	241	25	1

Cluster Statistics

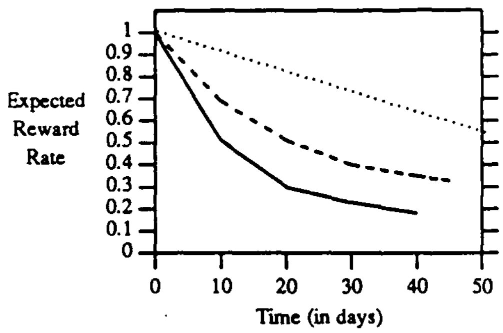
APPENDIX C



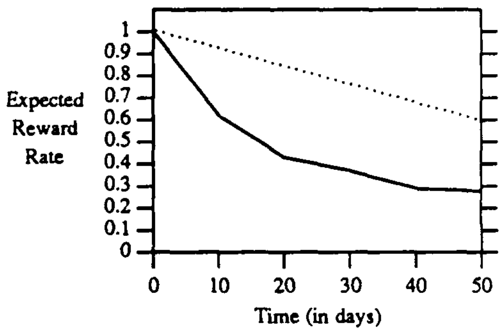
Earth



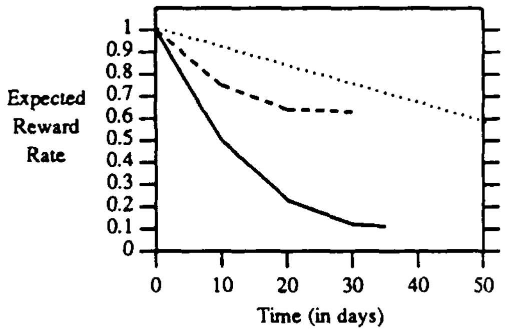
Europa



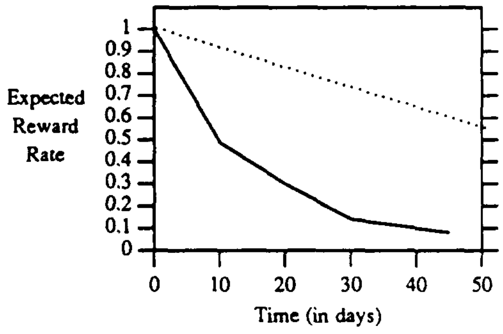
Jupiter



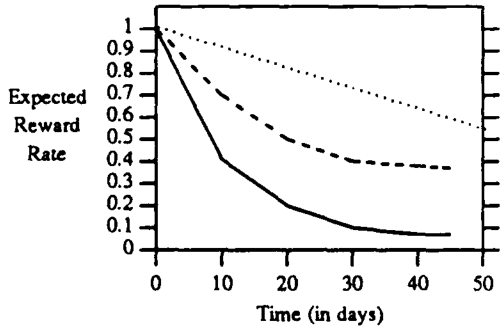
Leo



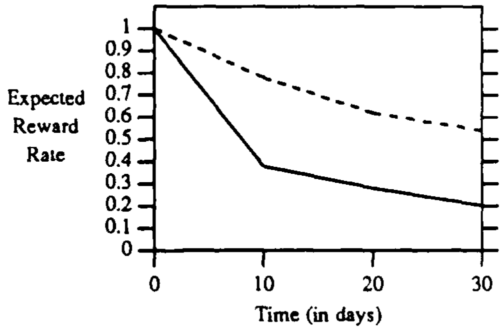
Mars



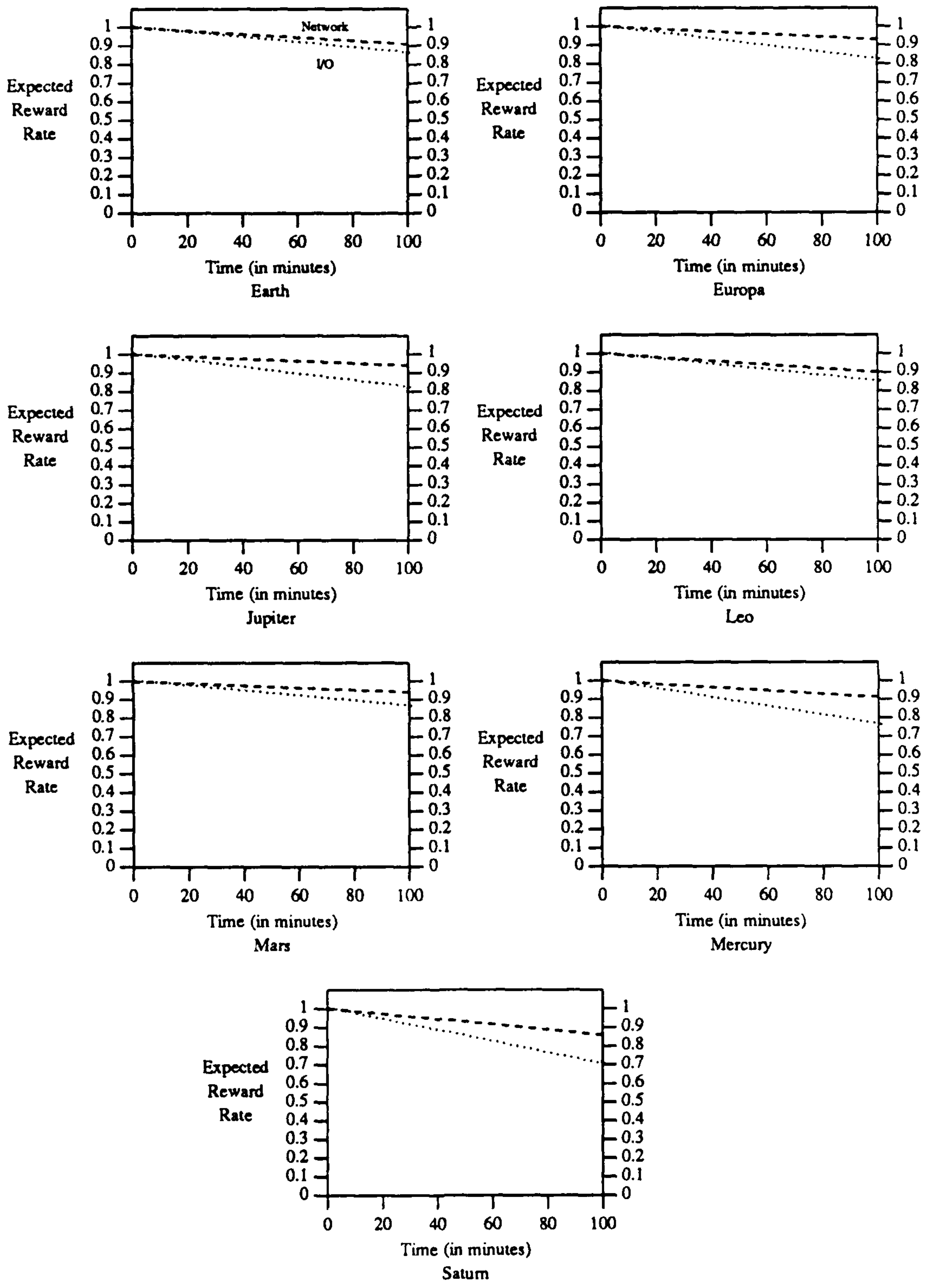
Mercury



Saturn



VaxCluster



Plots of Expected Reward Rate